情報理工学部　SN コース 2 回
セキュリティ・ネットワーク学実験 2
課題 5-3 レポート

2600200443-6
Yamashita Kyohei
山下 恭平

November 1 2021

1　概要

この実験では、Raspberry Pi と超音波レンジャー、ブザーを用いて、物が接近した際に音で警告するデバイスを設計した。このデバイスは、近年の自動車などによくみられる衝突回避システムや、ストーブなどの安全装置の基礎となるデバイスだと考えた。

2　外部仕様

2.1　開発対象の使い方に関する説明

プログラム実行中、センサは超音波を用いて距離を測定し続ける。その距離が一定の値よりも小さい時、ブザーがなるように設計した。実際には、自分の手を近づけたり、遠ざけたりすることで、距離を変化させ実験を行った。以下の図 1 は実際のデバイスの写真を撮ったものである。

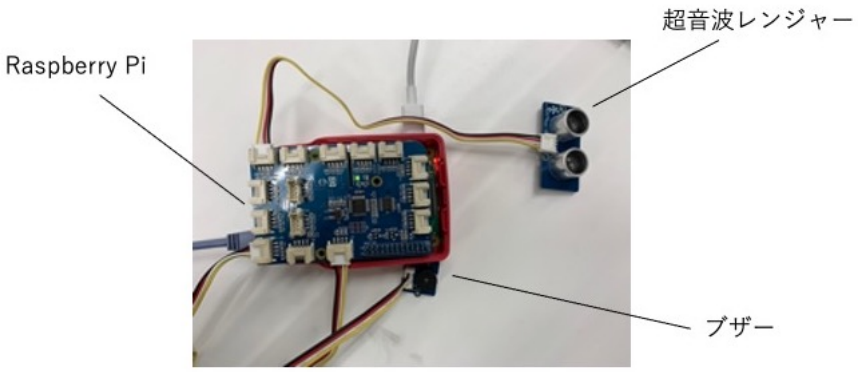

図 1　開発したデバイス

2.2　開発対象を構成するハードウェアと、その主な仕様

Raspberry Pi には Grove Starter Kit のシールドを利用し、各センサやアクチュエータを接続した。また、Raspberry Pi はセンサから情報の取得、アクチュエータの制御など、デバイスの主要部分ほとんどを担っている。超音波レンジャーは超音波により距離を測定しており、精度は 1cm 単位で測定可能である。ブザーは Raspberry Pi からの書き込み (0,1) によって on/off の切り替えが可能である。以下の表 1 はそれぞれのハードウェアを表にまとめたものである。

表 1　ハードウェア一覧

機器一覧	使用/情報
Raspberry Pi 4 Model B	センサ、アクチュエータを接続し、取得した値の処理、条件分岐を行う。
超音波レンジャー	超音波を送り、物体からのエコーを受信し、距離を測定。
ブザー	デジタル信号 (0,1) で on/off が切り替え可能。

2.3　ハードウェアやソフトウェアが担当する機能と、機能同士の関連

超音波レンジャーは距離を測定し、ブザーは音を鳴らすという役割がある。
この二つのハードウェアを制御しているのが Raspberry Pi であある。Raspberry Pi 上に書かれたプログラムは主に二つに分かれており。一つはセンサの値を取得する部分、もう一つは、その値が一定の値よりも小さいかどうかを判別する部分である。また、判別する部分では、その判別結果に応じた出力をブザーに送っており、一定の値以下であれば 1(ON) を、そうでなければ 0(OFF) をブザーへ出力するようになっている。
以下の図 2 は各機能の構成をまとめた図である。

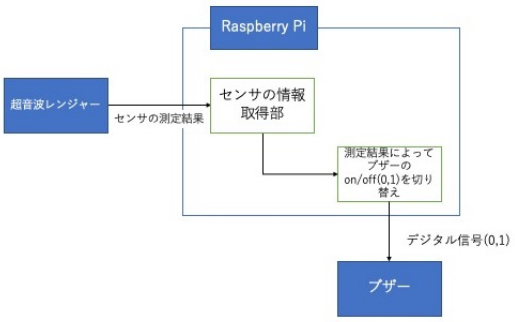

図 2　機能構成図

2.4　開発に用いたプログラミング言語と開発環境

今回の実験では、 実際に各センサやアクチュエータを制御するためのプログラムを、「Ju pyter Notebook」で「Python」言語を用いて開発した。
以下の表 2 は主な開発環境をまとめた表である。

表 2　まとめ図

主な開発環境	
OS	macOS Big Sur
開発環境	Jupyter Notebook
使用した言語	Python

3　内部仕様

3.1　各ハードウェアが備えるソフトウェアの詳細な設計

Raspberry Pi 上のプログラムで、各センサ、アクチュエータを制御するための関数が用意されている、超音波レンジャーでは ultrasonicRead() という関数があり、測定した距離が int 型で返ってくる。また、ブザーにおいては、digitalWrite() 関数で (0,1) を書き込むことで、ブザーの on/off を切り替えている。以下は、変

数、関数をまとめた表である

表 3　変数/関数 まとめ表

変数名	説明
ranger	超音波センサのポート番号を指定するための変数。
buzzer	ブザーのポート番号を指定するための変数。
sensor_value	センサから取得した値を格納する変数。

関数名	説明
ultrasonicRead()	超音波センサが距離を測定するための関数。
digitalWrite()	ブザーへデジタル信号を書き込む関数。
pinMode()	指定された入出力ポートに対して、入力か出力かを設定する関数。

3.2　各ハードウェアが備えるソフトウェアにおける処理の流れ

このデバイスは、初めにセンサの値を取得し、その取得した結果を判定し、判定の結果に応じてブザーに信号を送る。といった流れを無限ループで回すことによって実現している。
図 3 はその様子をフローチャートにまとめたものである。

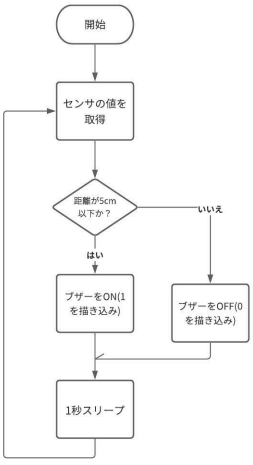

図 3　フローチャート

4　実行例

Jupyter Notebook 上でプログラムを実行した後に、超音波レンジャーに手を近づけてから、遠ざけるという一連の動作を行ったところ、手の距離が 5cm あたりのところからブザーがなりはじめた。
図 4 は 1 秒おきのセンサの測定結果と、ブザーの状態をプリントした Jupyter Notebook 上のスクリーン

ショットであり。図 5 は実際に超音波レンジャーに手を近づける様子を写真に撮ったものである。

```
sensor_value:8
buzzer:OFF
sensor_value:6
buzzer:OFF
sensor_value:4
buzzer:ON
sensor_value:2
buzzer:ON
sensor_value:3
buzzer:ON
sensor_value:6
buzzer:OFF
sensor_value:7
buzzer:OFF
```

図 4　実行例 1

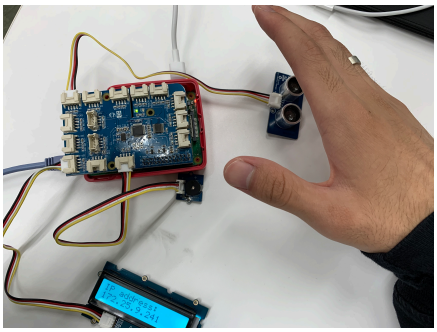

図 5　実行例 2

5　ソースコード

以下はソースコードである。
超音波レンジャーで測定した距離を 20 行目で取得し、取得した値を 25 行目から判別し、ブザーの on/off を切り替えている。この全体の流れを 16 行目の while 分で無限ループにしている。

Listing 1　5-3.py

```
1
2     import time
3     from grovepi import *
4
5     # A4ポートへ超音波センサを接続
6     ranger = 4
7
8     # D5ポートへブザーを接続
9     buzzer = 5
10
11    # センサは入力、ブザーは出力として設定
12    pinMode(ranger,"INPUT")
13    pinMode(buzzer,"OUTPUT")
14    time.sleep(1)
15
16    while True:
17        try:
18
19            # センサの値を読み込み
20            sensor_value = ultrasonicRead(ranger)
21
22            print("sensor_value:%d" %(sensor_value))
23
24            # 5以下ならブザーON
25            if sensor_value < 5:
```

```
26                 print("buzzer:ON")
27                 digitalWrite(buzzer,1)
28             else:
29                 print("buzzer:OFF")
30                 digitalWrite(buzzer,0)
31 
32             time.sleep(1)
33 
34         except KeyboardInterrupt:
35             break
36 
37         except IOError:
38             print ("Error")
```